LARSOFT OPTICAL SIMULATIONS UPDATE

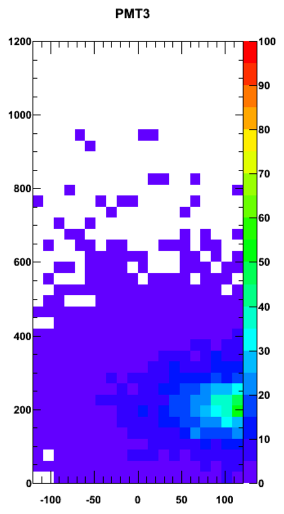
Christie Chiu and **Ben Jones**

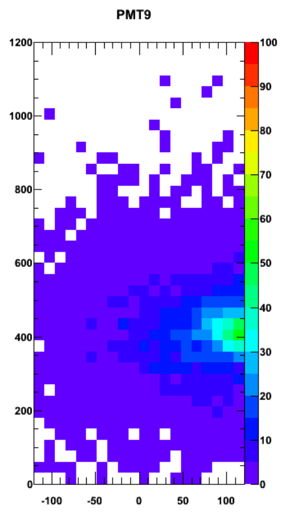
Massachusetts Institute of Technology

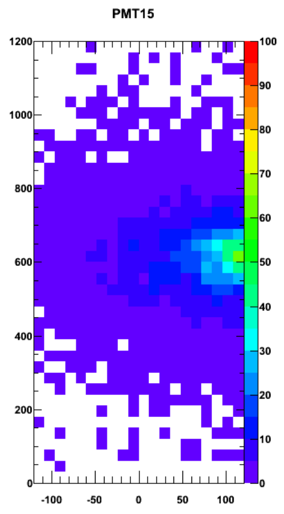
This is not an overview talk…

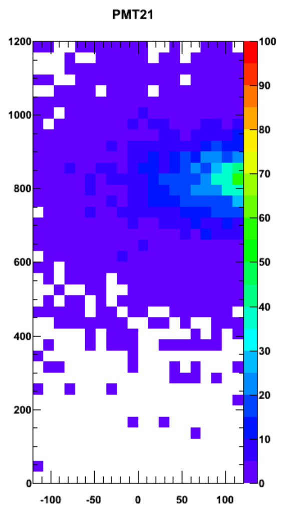
For a long discussion of the LArSoft optical simulation packages, see my talk from a few weeks ago to this meeting:

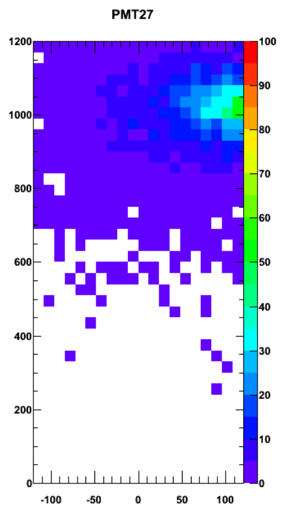
- https://cdcvs.fnal.gov/redmine/documents/495











Recent Progress (since overview talk)

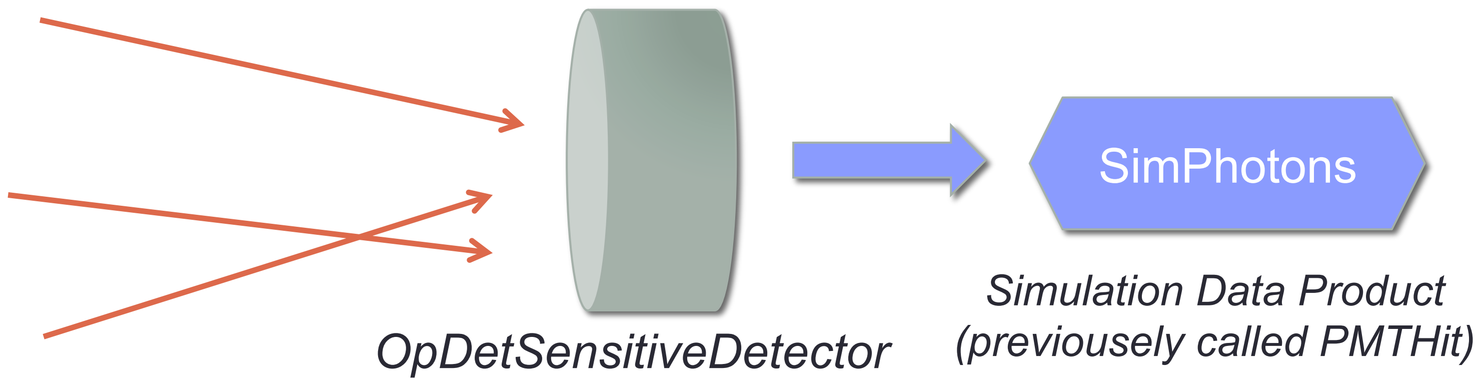
- Updates to optical geometry handling
- Fast / lookup table based optical simulation
- Optical raw data / signal processing
- Optical subevent finding and T0 determination algorithms

Detector Agnosticism

- All LArSoft optical code has always been easily transferrable to an LBNE type geometry and optical system.
- However, to make it even clearer, now everything previously named "PMT____" in LArSoft is named "OpDet____".
- I can assist an LBNE person or people in connecting the relevant sensitive objects up in the LBNE geometry to run optical simulations in LArSoft.

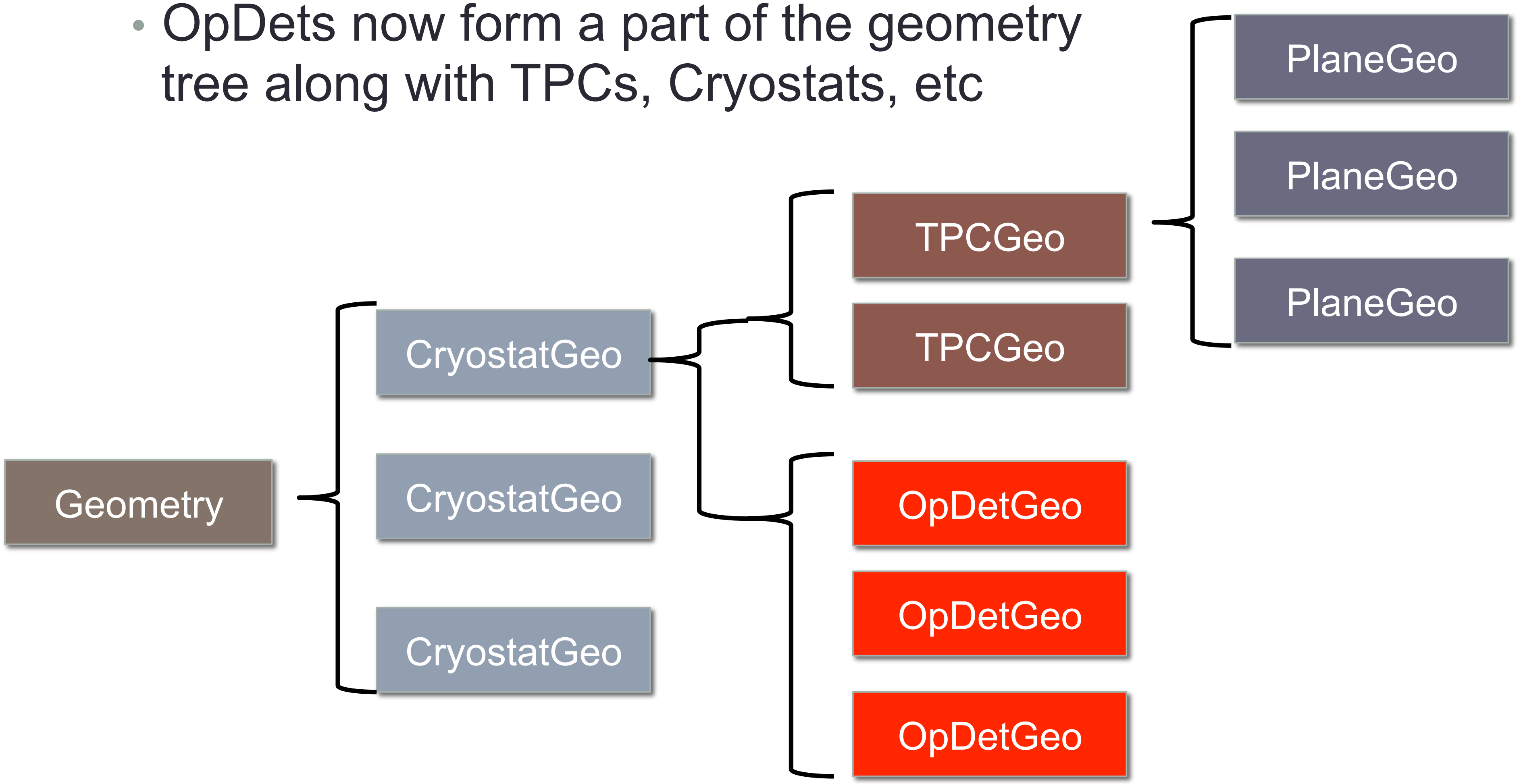
Geometry Changes

- In previous implementation, LArG4 found sensitive elements by specified name in gdml file, and gave each an ID.
- This meant that volumes and ID's were not accessible outside LArG4.
- This now globalized, in the sense that Geometry now controls ID assignments and knows about OpDet positioning
- The mapping between Geometry and LArG4 objects is handled by the LArG4/OpDetLookup object.



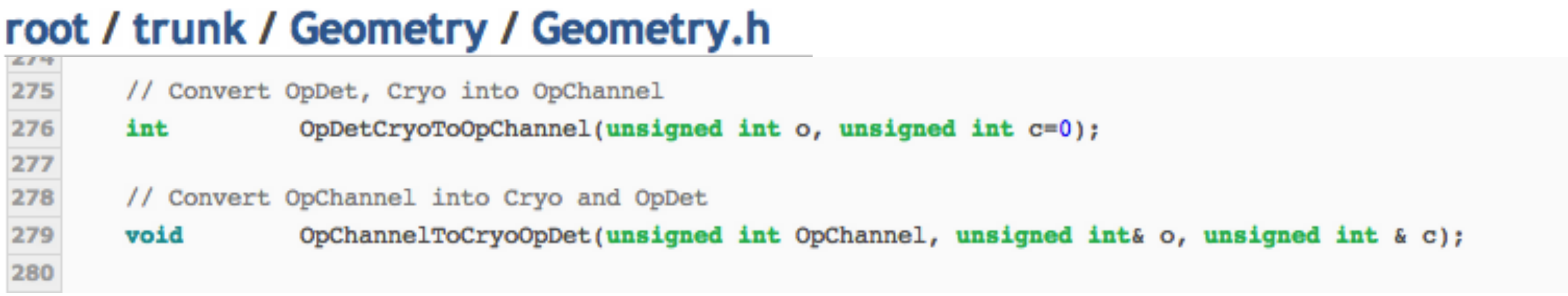
geo::OpDetGeo

- OpDets now form a part of the geometry tree along with TPCs, Cryostats, etc



Labeling Convention

- OpDets exist within a given cryostat
- Similarly to Wires, every OpDet has a unique ID, OpChannel
- Each also has an address (c,o), Cryostat (o to N_cryo) and OpDet (0 to N_opdet)
- Methods added to geometry to facilitate conversion between the two schemes, like for wires and planes.
- Everything stored in the event is indexed by OpChannel only.



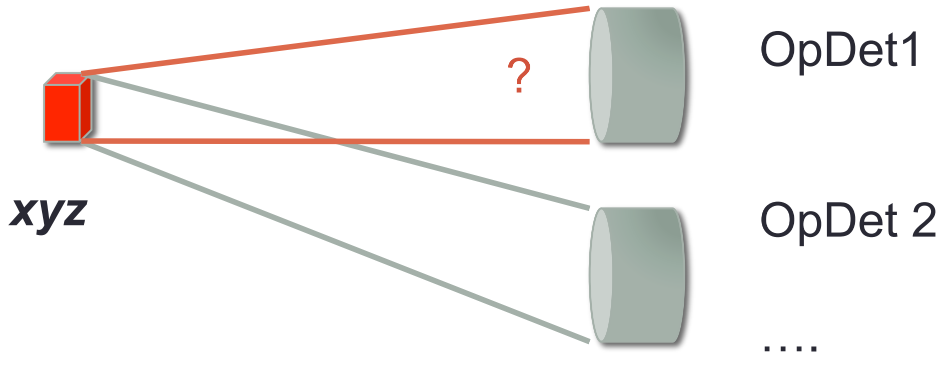
root / trunk / Geometry / Geometry.h

```
    // Convert OpDet, Cryo into OpChannel
    int           OpDetCryoToOpChannel(unsigned int o, unsigned int c=0);

    // Convert OpChannel into Cryo and OpDet
    void          OpChannelToCryoOpDet(unsigned int OpChannel, unsigned int& o, unsigned int & c);
```

PhotonVisibilityService

- Service added to PhotonPropagation package to facilitate optical reconstruction algorithms
- Main function: Given a point xyz in the detector, how likely is it that 1PE produced there will be seen at each OpDet?

- Two modes of operation :
  - 1) **Simple** – 1/r^2 and solid angle to opdet surface
    - (no reflections, etc)
  - 2) **Library** – use fastsim library (tbi)
    - (full reflections, scattering, etc accounted for. But requires filled library)

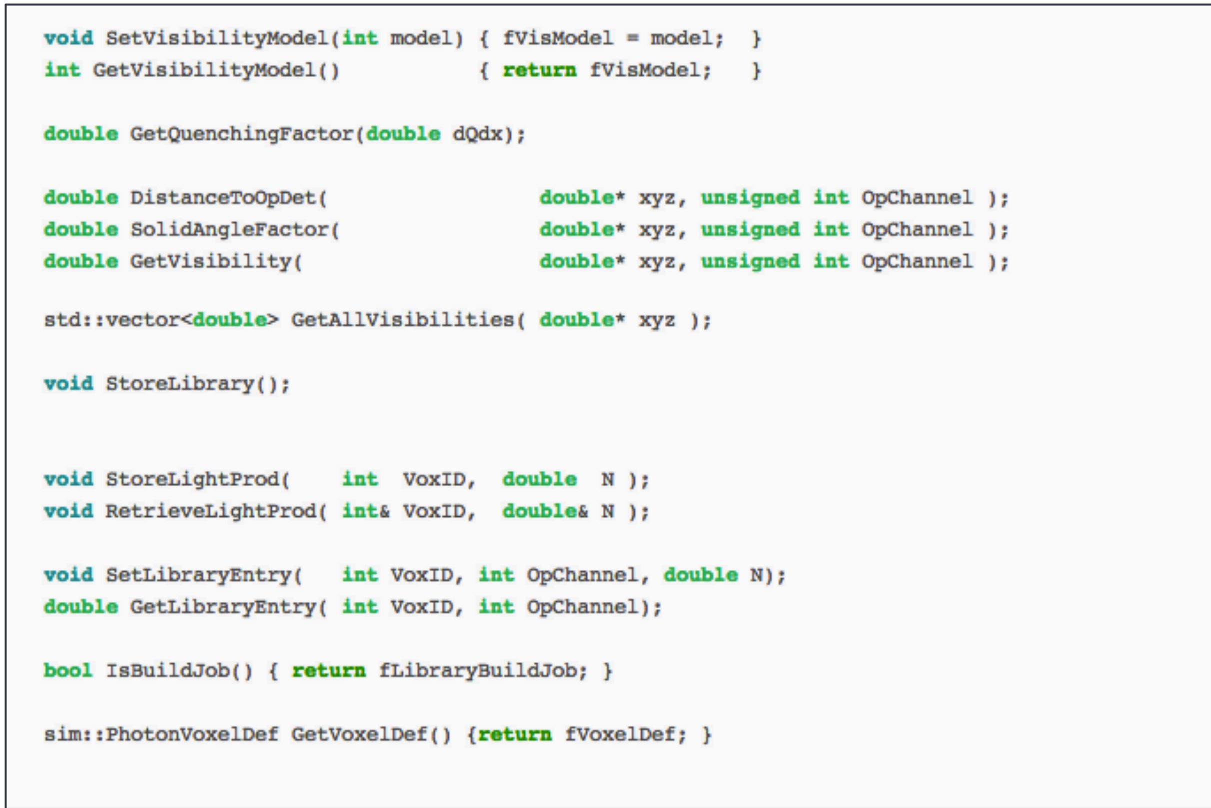


PVS: Library Mode

- This service has now subsumed the old “PhotonLibraryService”, and handles all lookup based optical simulation methods

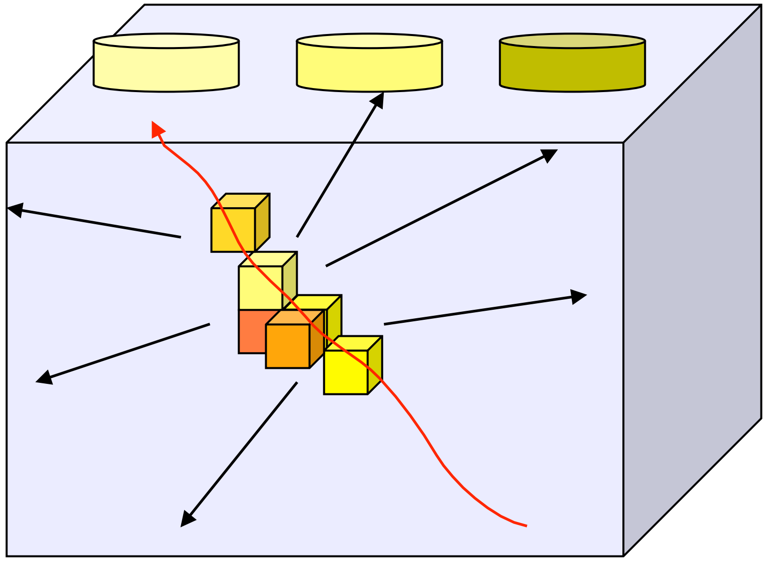
- As such these methods are now available to both simulation and reconstruction algorithms.

- This represents a MAJOR overhaul of the library building and sampling since the last talk



```
void SetVisibilityModel(int model) { fVisModel = model;  }
int GetVisibilityModel()           { return fVisModel;   }

double GetQuenchingFactor(double dQdx);

double DistanceToOpDet(                  double* xyz, unsigned int OpChannel );
double SolidAngleFactor(                 double* xyz, unsigned int OpChannel );
double GetVisibility(                    double* xyz, unsigned int OpChannel );

std::vector<double> GetAllVisibilities( double* xyz );

void StoreLibrary();


void StoreLightProd(    int  VoxID,   double  N );
void RetrieveLightProd( int& VoxID,   double& N );

void SetLibraryEntry(   int VoxID, int OpChannel, double N);
double GetLibraryEntry( int VoxID, int OpChannel);

bool IsBuildJob() { return fLibraryBuildJob; }

sim::PhotonVoxelDef GetVoxelDef() {return fVoxelDef; }
```

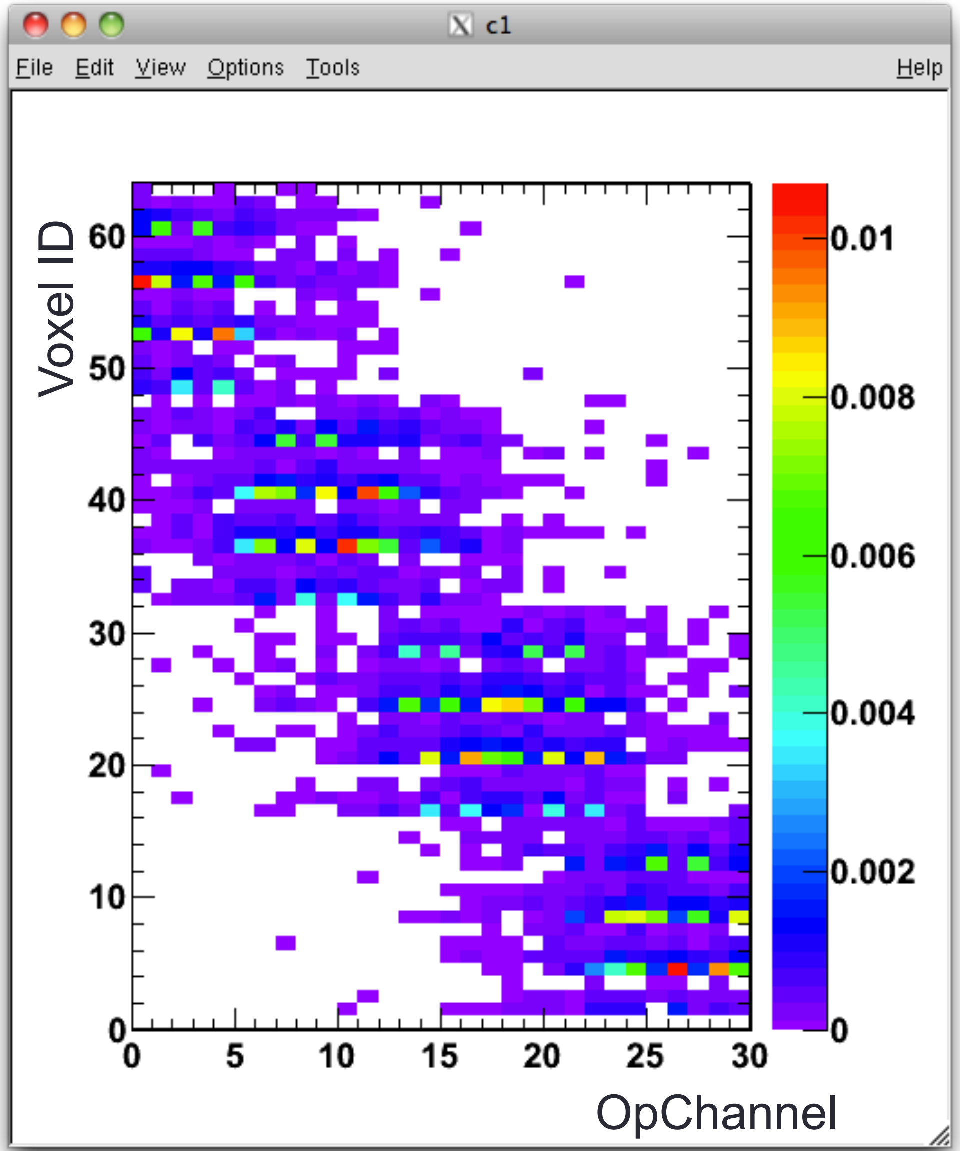
Library Building



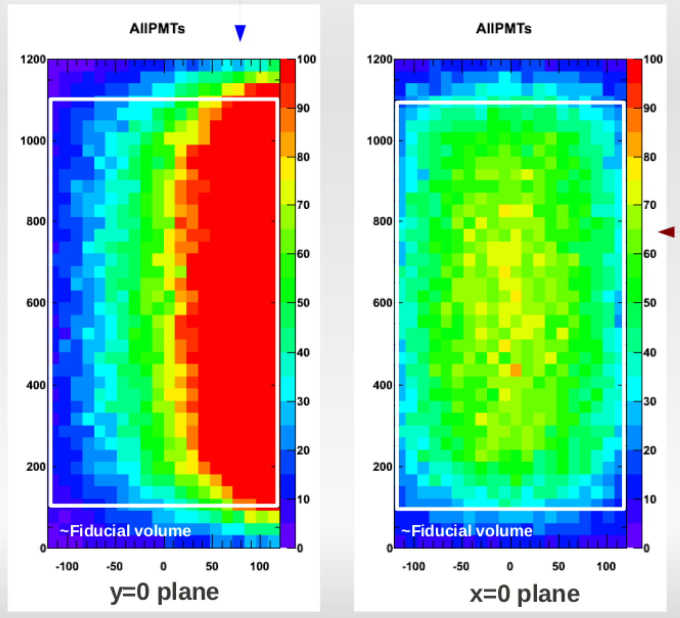
- Enable service PhotonVisibilityService
- Specify in its configuration:
  - *BuildLibrary : true*
  - *Number of voxels in x,y,z*
  - *Optional : sub-region of detector to simulate*
- Run modules LightSource, LArG4, SimPhotonCounter
- Brightness should be specified for LightSource. Other than this, PhotonVisibilityService will configure the modules appropriately.
- Library file is output through the TFileService

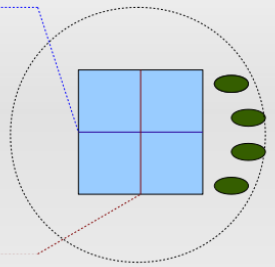
Library Build Test Run

- Quick test run : 4x4x4 voxels, 10,000 photons in each
- Library build job takes about 10 mins



Colors show fraction of photons detected

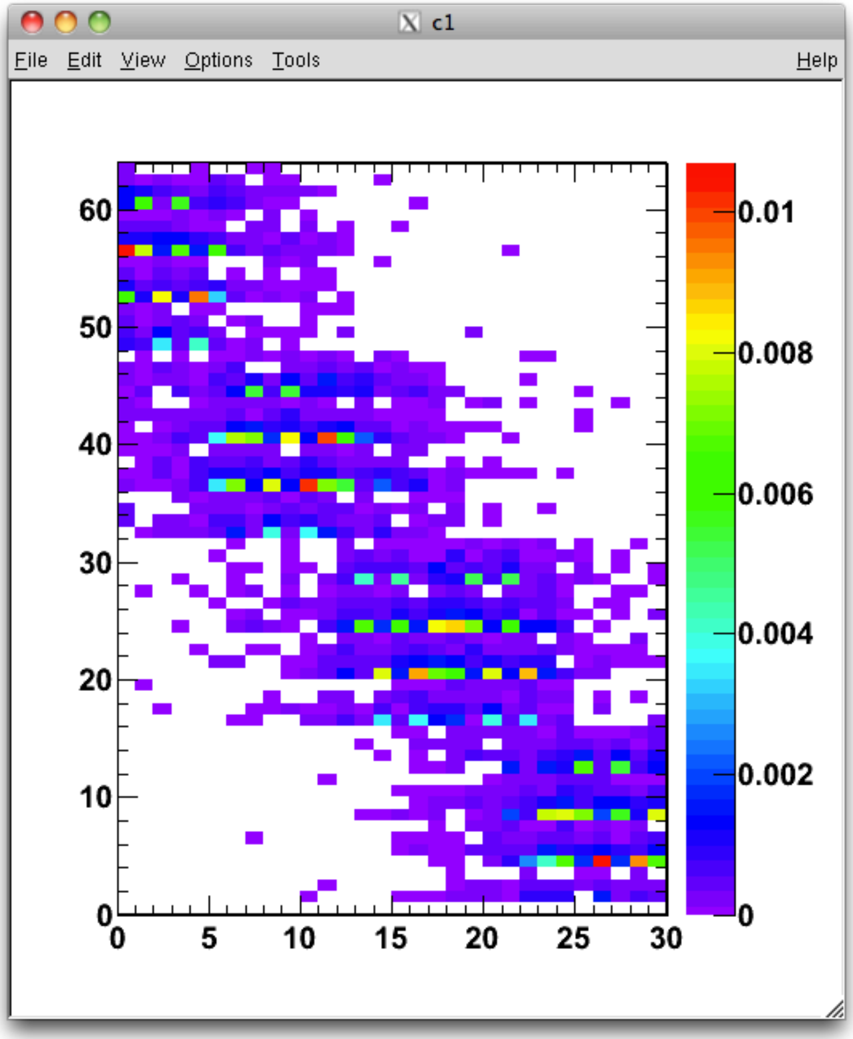




Number of photoelectrons summed over all PMTs for point source of **5MeV** equivalent at different detector points

Trigger should be possible on 1 p.e. We are sensitive at all points on these plains within fiducial volume.

Effectively the same information, but broken down per PMT and in a weird coordinate system.



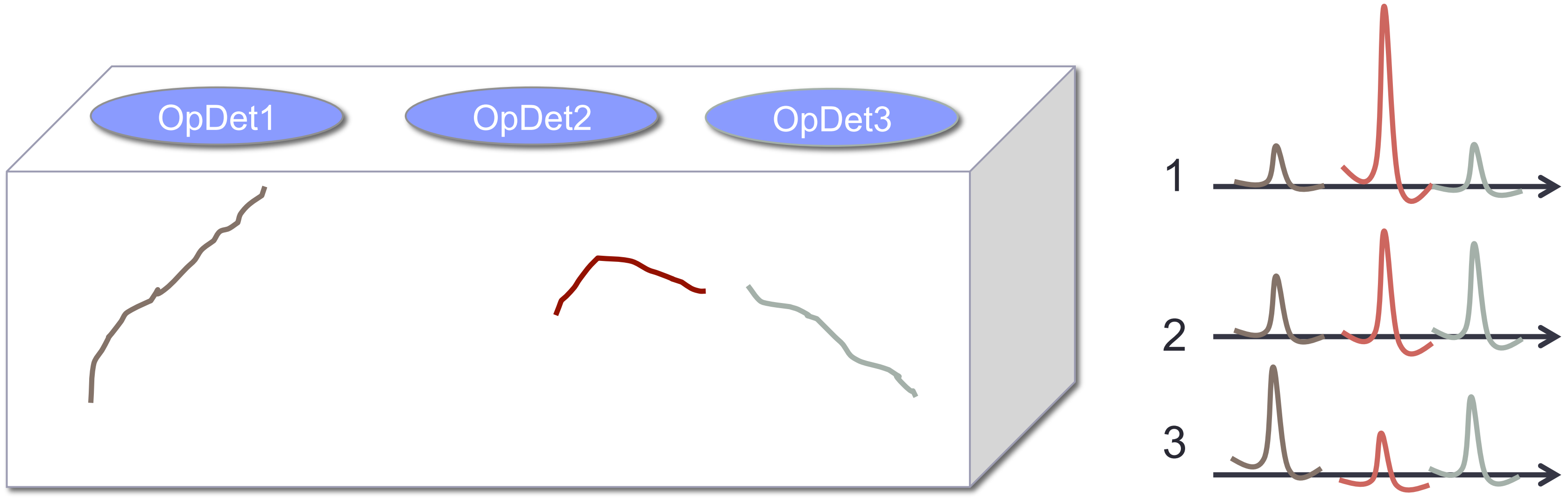
Library sampling simulation tool coming online soon (most code exist in uncommitted form).

Scaling Up to Grid Job

- Optimistically : later this week to build full MicroBooNE library using grid jobs.
- Do a few voxels in each and then combine library library files offline.
- Feasible library size for MicroBooNE light collection:
  - 25 x 25 x 100, 100,000 photons / vox ~ 10 x 10 x 10 cm voxels
- Scaling up, this is about 1600 hours of grid jobs.
- These numbers not optimized, and there may be room for increasing efficiency by disabling irrelevant physics in LArG4.
- LBNE voxel scale can be debated. Maybe cm is not necessary?

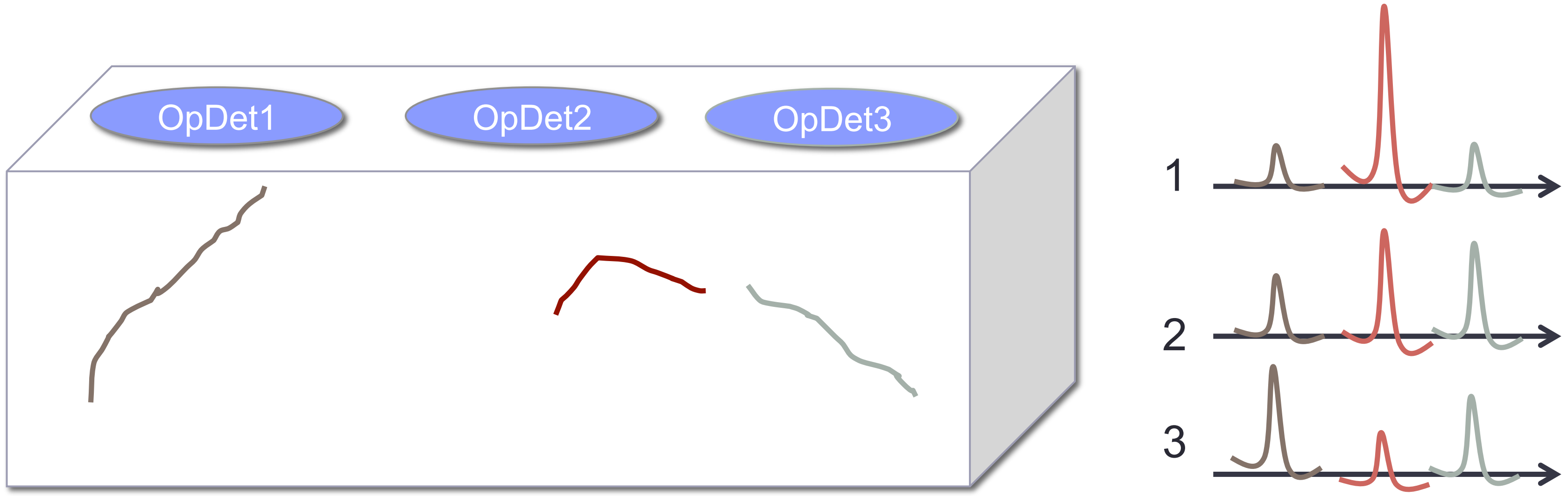
Geometrical T0 Finding

- 1: Find subevents by matching large PMT signals in time
- 2: Make hypotheses of relative amount of light per PMT for each track
- 3: Likelihood fit to match track to light hypothesis, and find T0.



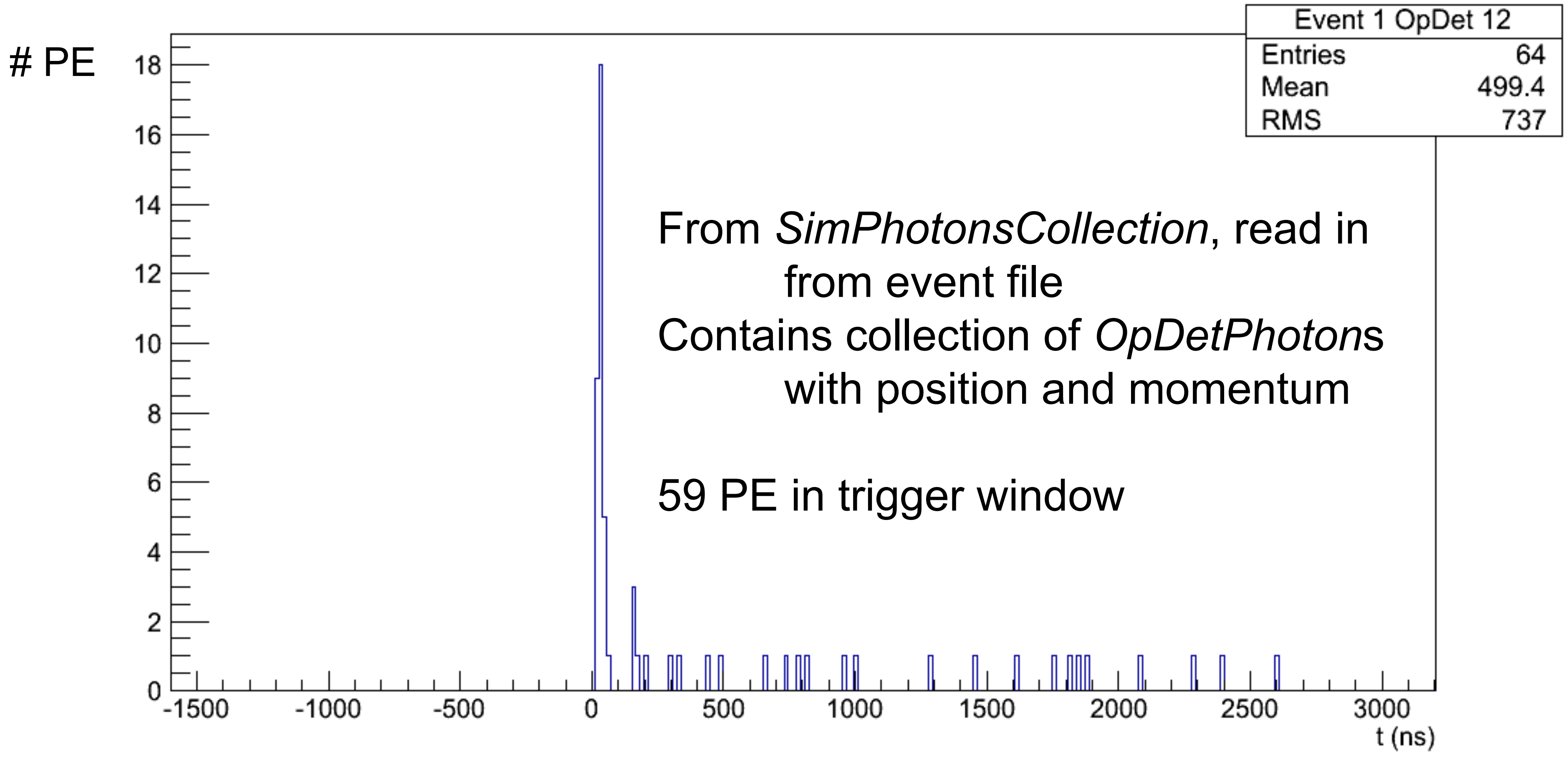
Geometrical T0 Finding

- **1: Find subevents by matching large PMT signals in time**
- **2: Make hypotheses of relative amount of light per PMT for each track**
- 3: Likelihood fit to match track to light hypothesis, and find T0.



OpDet digitized signal simulation

Photon arrival times



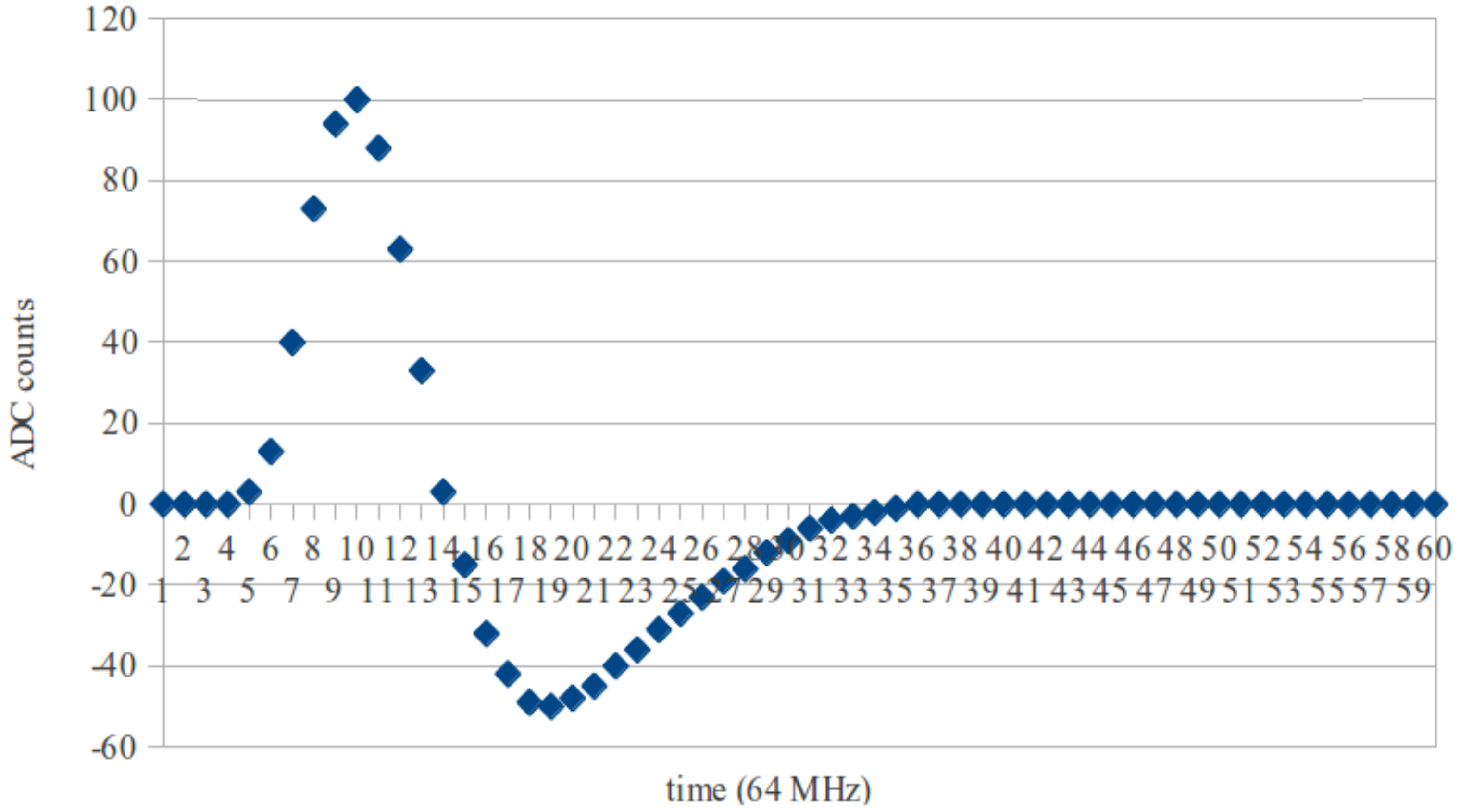
From *SimPhotonsCollection*, read in from event file

Contains collection of *OpDetPhotons* with position and momentum

59 PE in trigger window

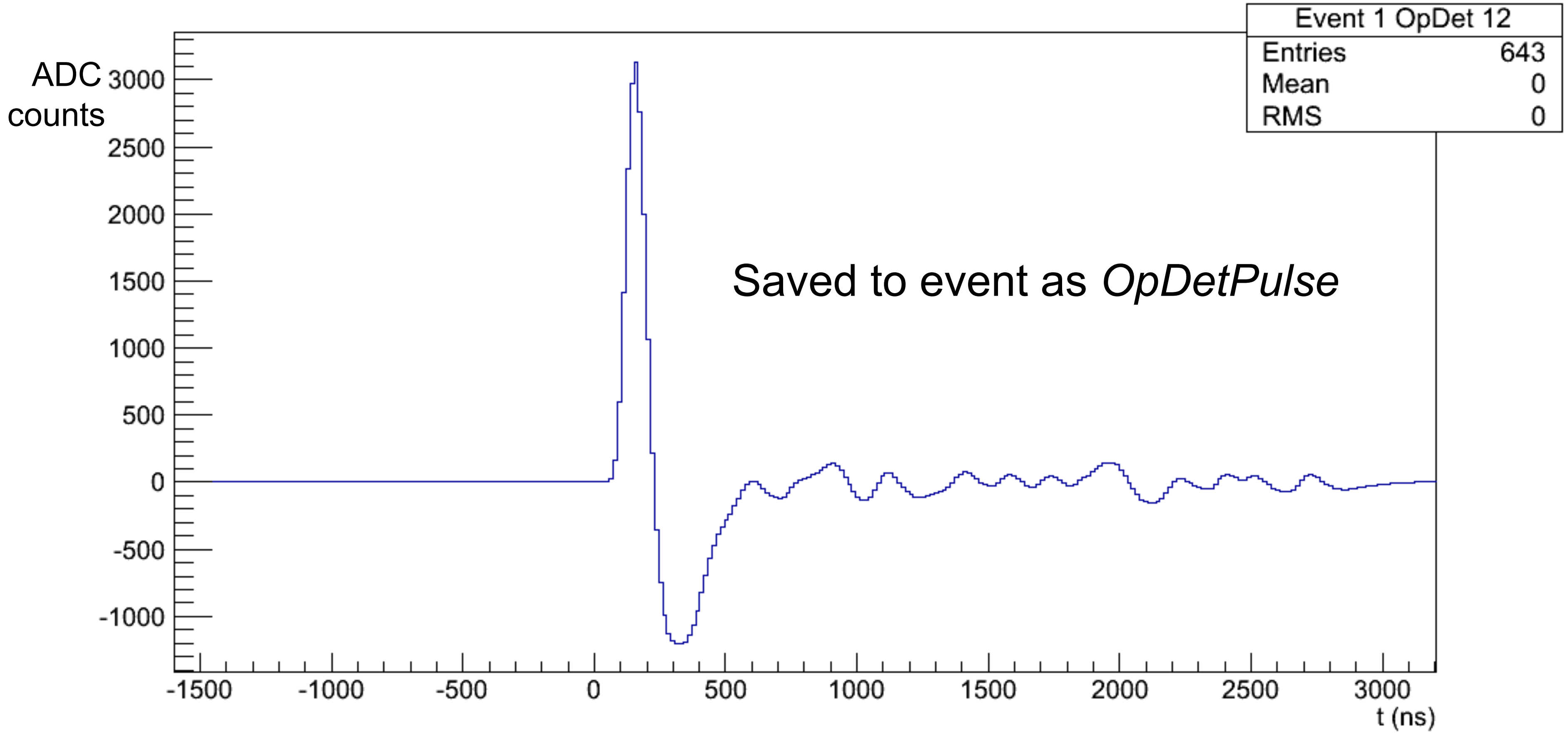
OpDet digitized signal simulation

Bare 1 PE signal (input parameter)



OpDet digitized signal simulation

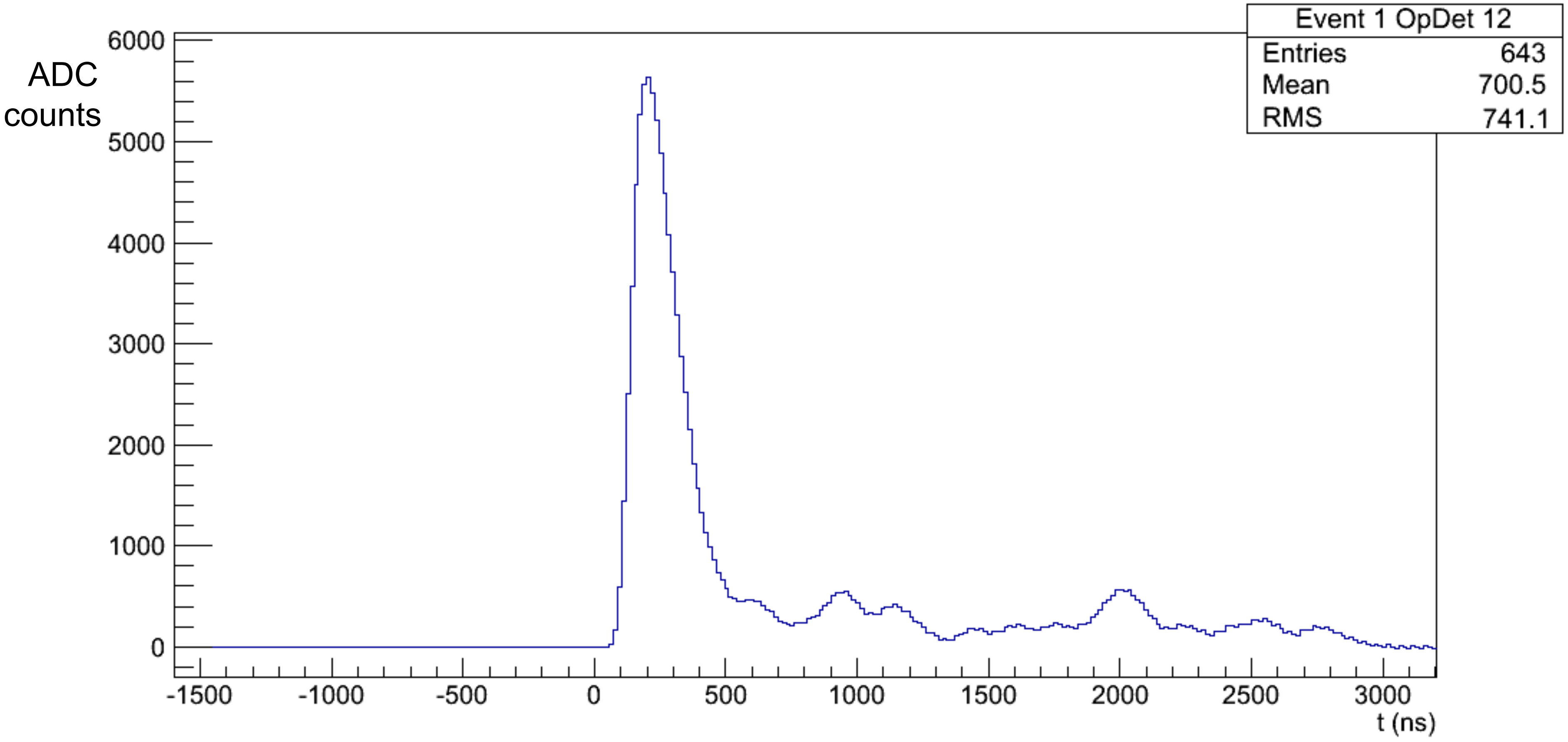
Digitized PMT signal



Saved to event as *OpDetPulse*

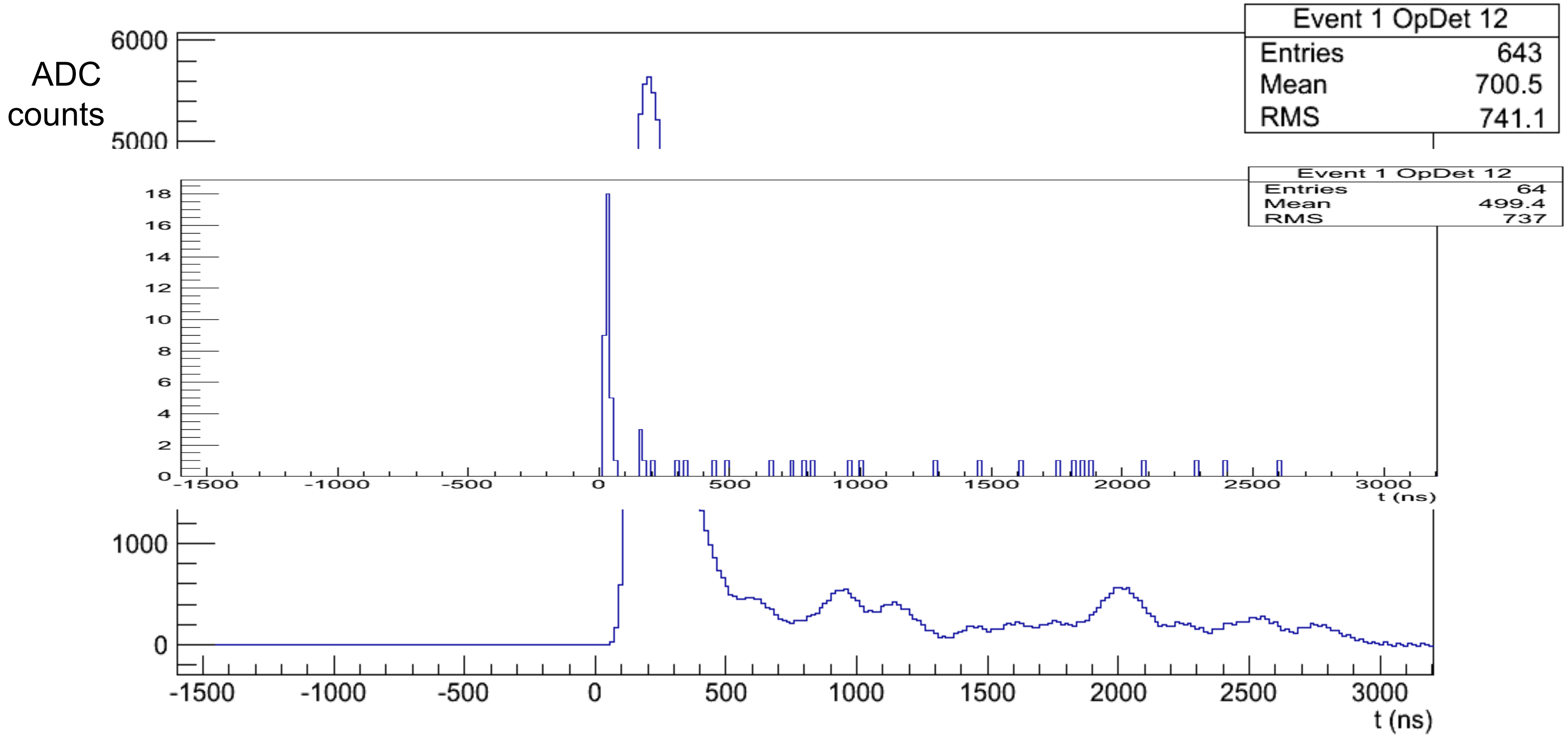
Signal deconvolution

$\tilde{a}_n = a_n - a_{n-3}$



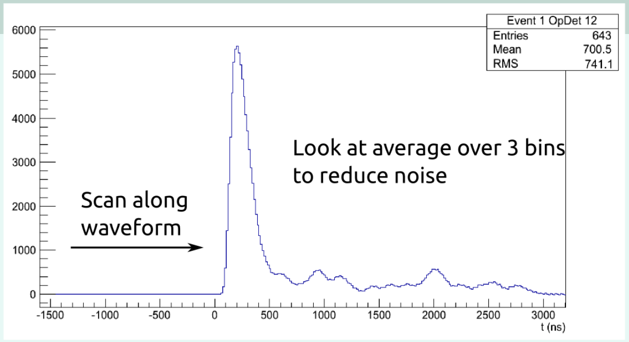
Signal deconvolution

Notice peaks line up with photon arrival times



Gaussian fits

Find fit regions for each peak



looking for minimum
(end of fit range *i*,
beginning of fit range *i+1*)

Increase in amplitude → Save end of fit range *i*, start of fit range *i+1*

looking for maximum
(of fit range *i*)

Observed "shoulder"behavior:
extra inflection point

Save end of fit range *i*,
start of fit range *i+1*

Decrease in amplitude
Maximum > 1 PE pulse height

Cleanup incomplete fit range at end of trigger window:
If we've seen a maximum, set last point as end of fit range
Otherwise, discard current fit range

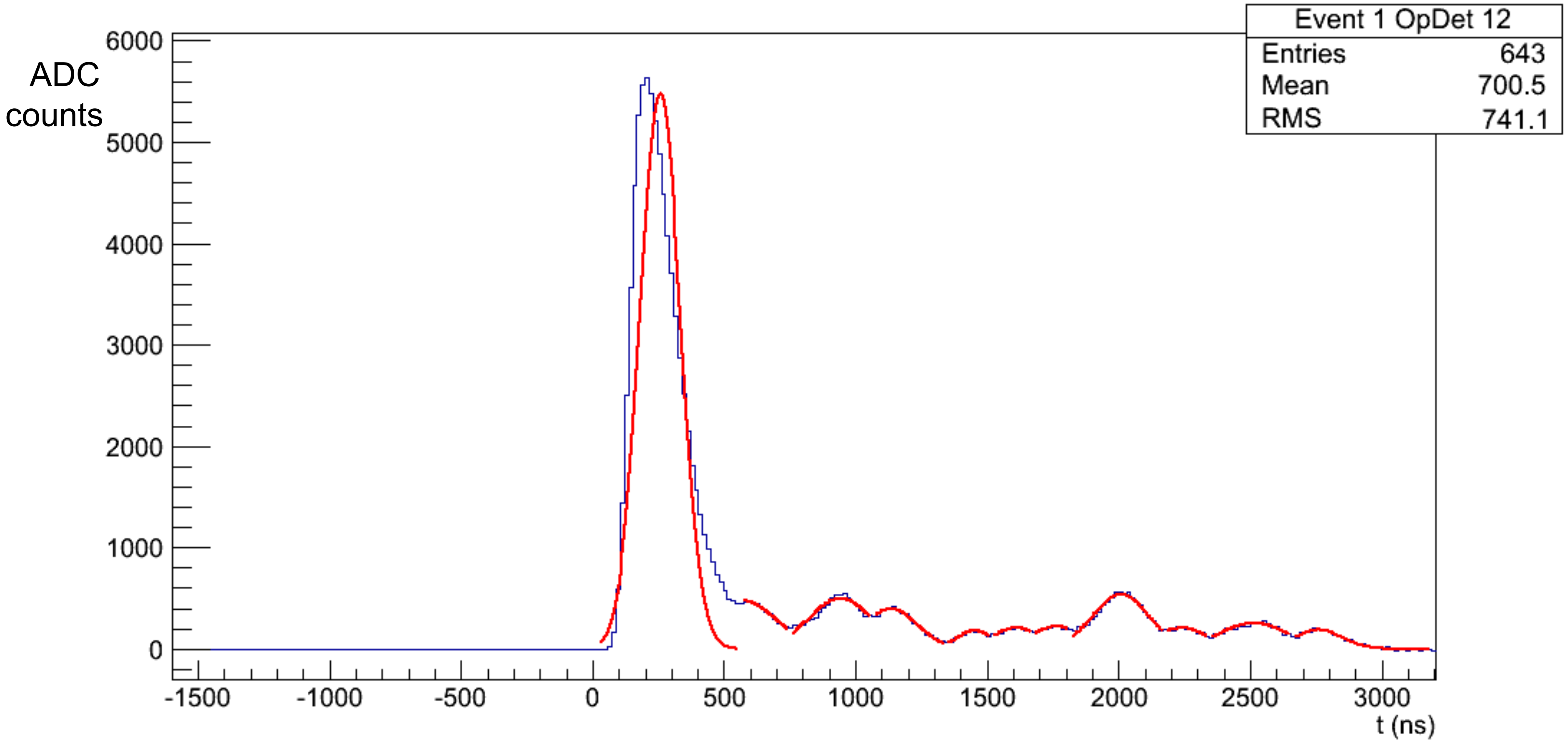
Perform fits to each peak separately

Use fit parameters from above as seeds

Perform 1 fit to total waveform

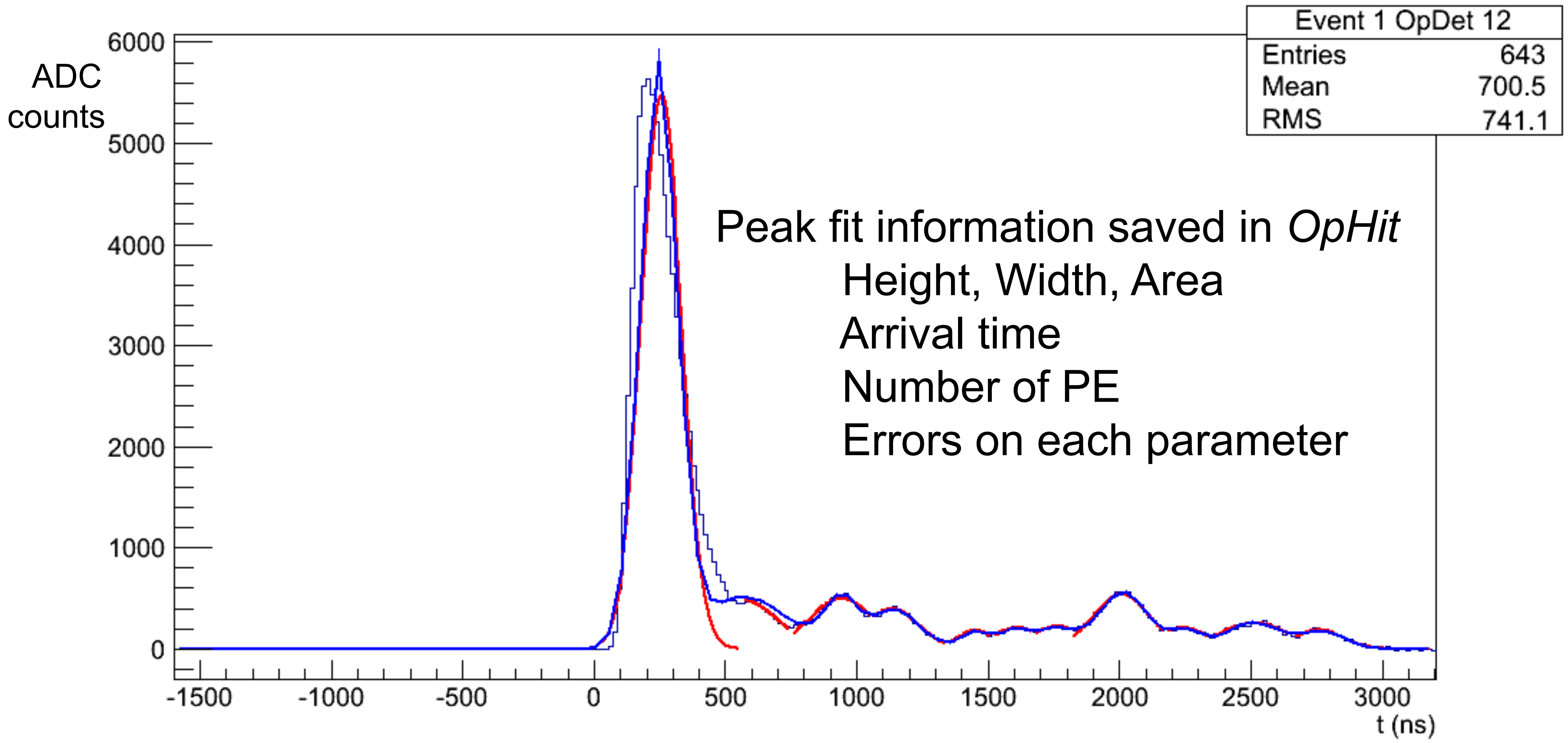
Gaussian fits

Preliminary 1-peak fits



Gaussian fits

Preliminary 1-peak fits & Final *n*-peak fit

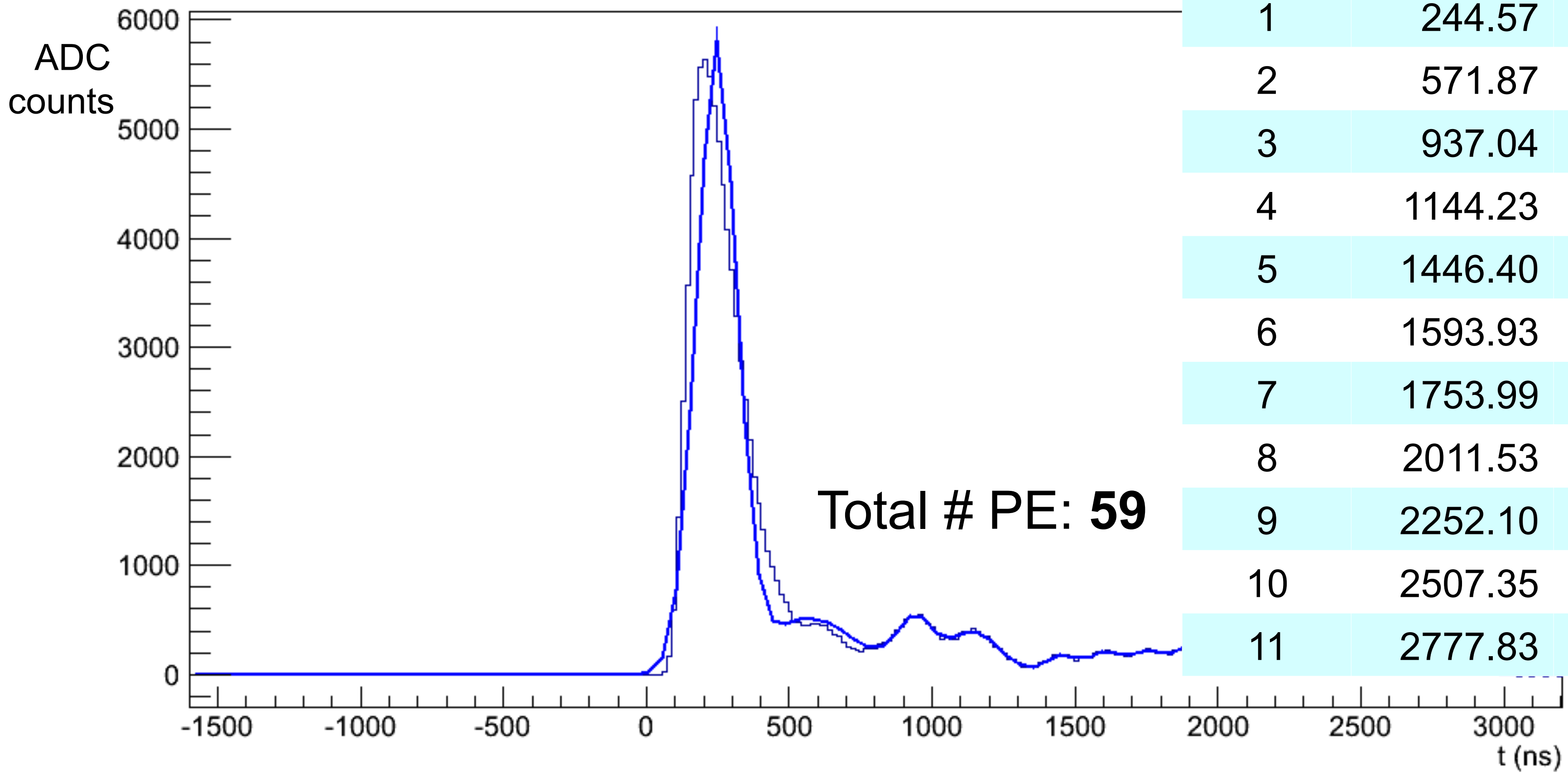


Peak fit information saved in *OpHit*

- Height, Width, Area
- Arrival time
- Number of PE
- Errors on each parameter

Data extraction

Use pulse area to find # of PE



Total # PE: **59**

Peak #	Time (ns)	# PE
1	244.57	35
2	571.87	7
3	937.04	3
4	1144.23	3
5	1446.40	1
6	1593.93	1
7	1753.99	1
8	2011.53	4
9	2252.10	1
10	2507.35	2
11	2777.83	1

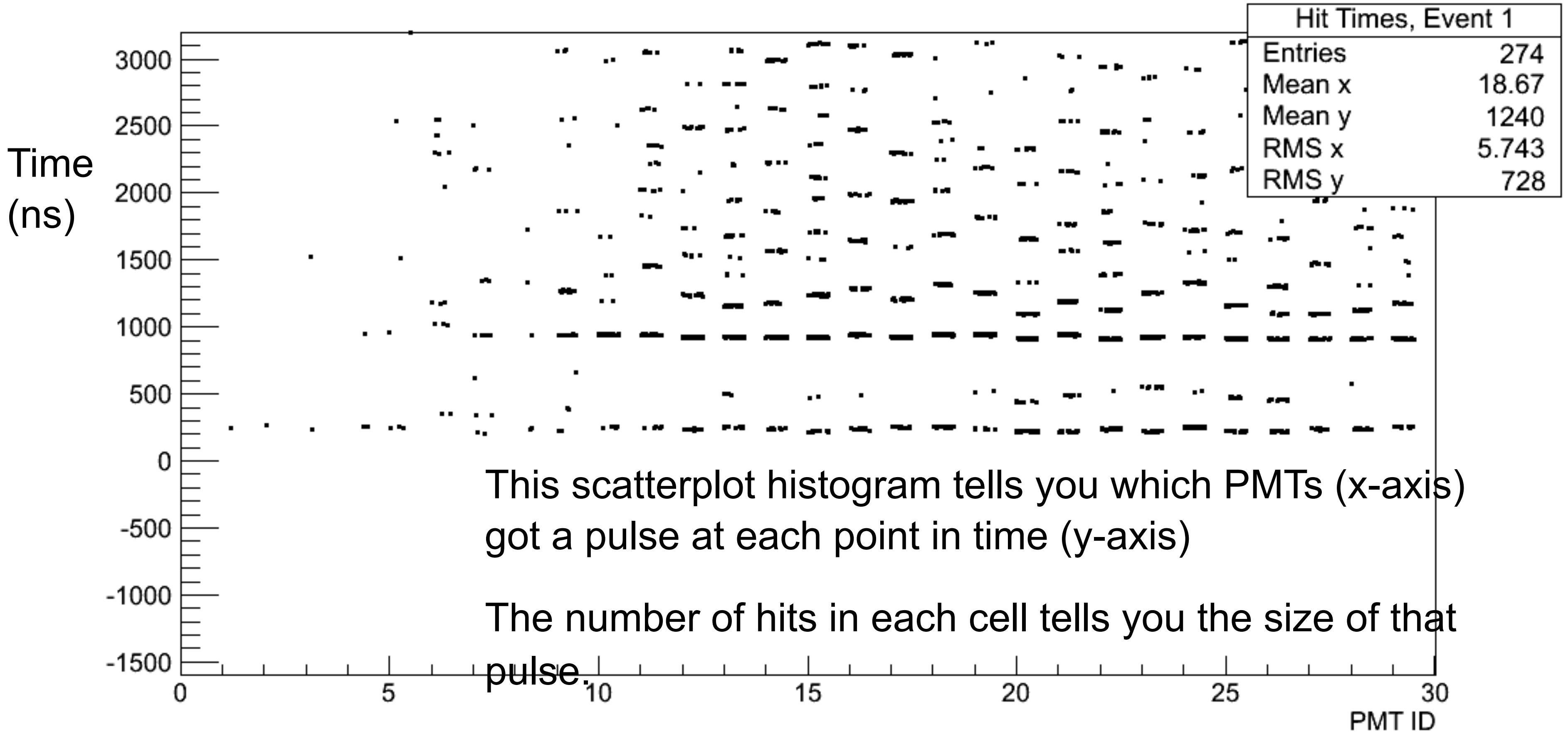
Data extraction

Accuracy:

- PE counts are off by ±3 at most
- 2.6% error

Tested on 66 signals so far

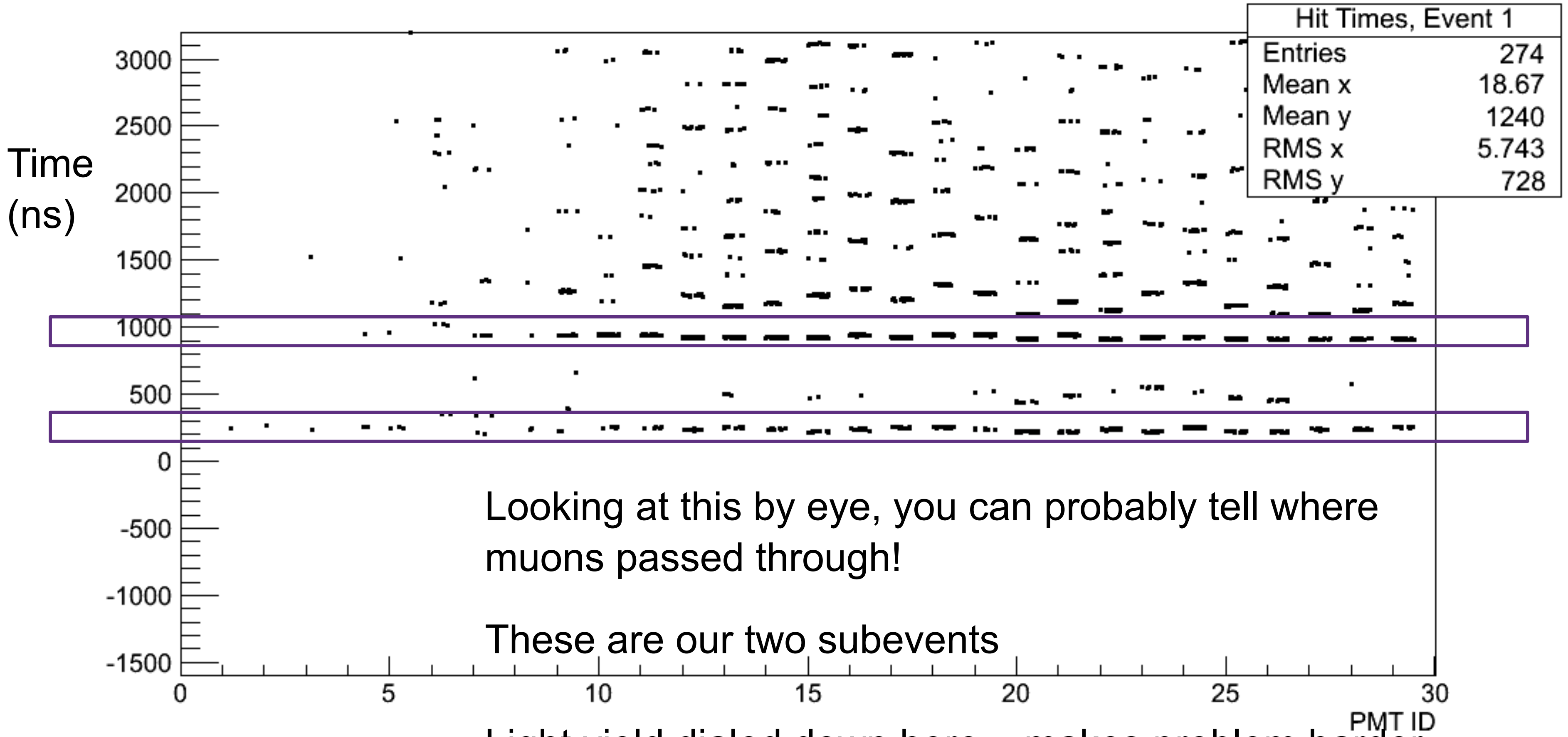
Optical Subevent Finding



This scatterplot histogram tells you which PMTs (x-axis) got a pulse at each point in time (y-axis)

The number of hits in each cell tells you the size of that pulse.

Optical Subevent Finding



Looking at this by eye, you can probably tell where muons passed through!

These are our two subevents

Light yield dialed down here – makes problem harder, not easier

The algorithm

1) Compute average # & standard deviation of PE in peaks for each PMT
   - - We will only consider peaks larger than (avg – 1 stdev)
2) Find the largest peak
3) Scan through all peaks and find ones that fit in time with this largest peak
   - - Each peak can be in *at most* one subevent
4) Find the next largest peak not in a subevent, and repeat until all peaks have been sorted
5) Merge subevents that are very close in time and have completely disjoint sets of PMTs listed
   - - Each subevent has at most 1 peak from any given PMT

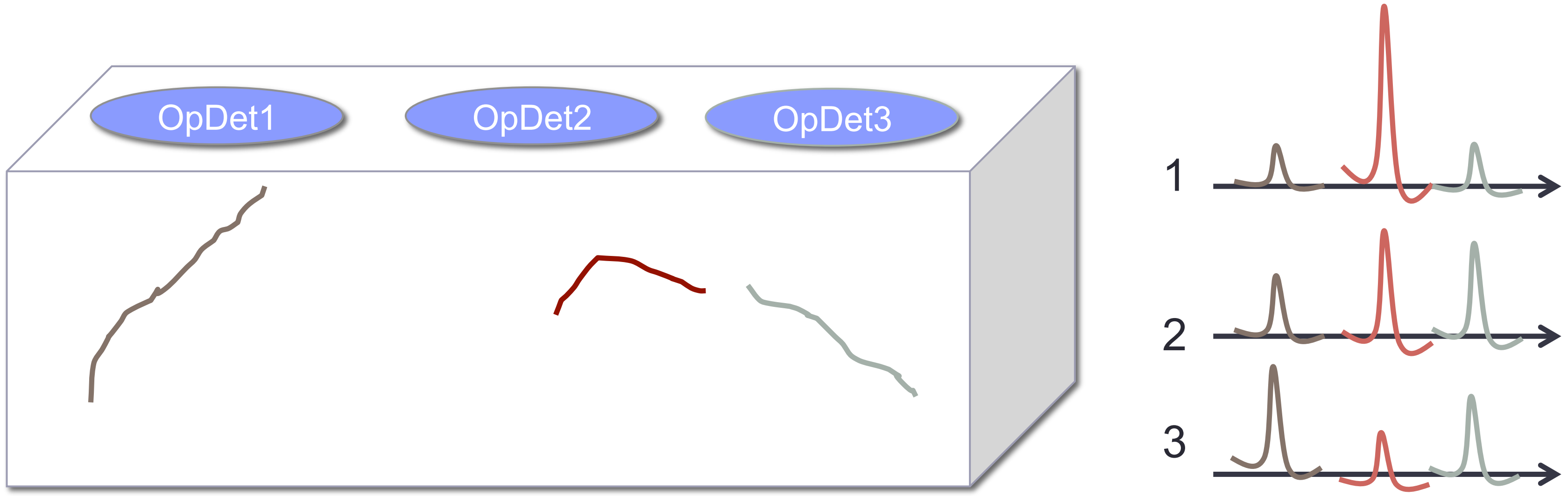
Accuracy

- Finds correct number of subevents for:
  - Up to 6 particles (muons), sometimes 7
  - Particle separation times > 180 ns
    - At smaller separation times, peaks merge
  - All these particles are traveling straight through detector volume (seen by most PMTs)
- Subevent timing fluctuates by about 100 ns (if we run same event multiple times)
  - Fit parameters waver slightly

- Now waiting on fast sim to be ready in order to test efficiency over different event classes – update next meeting

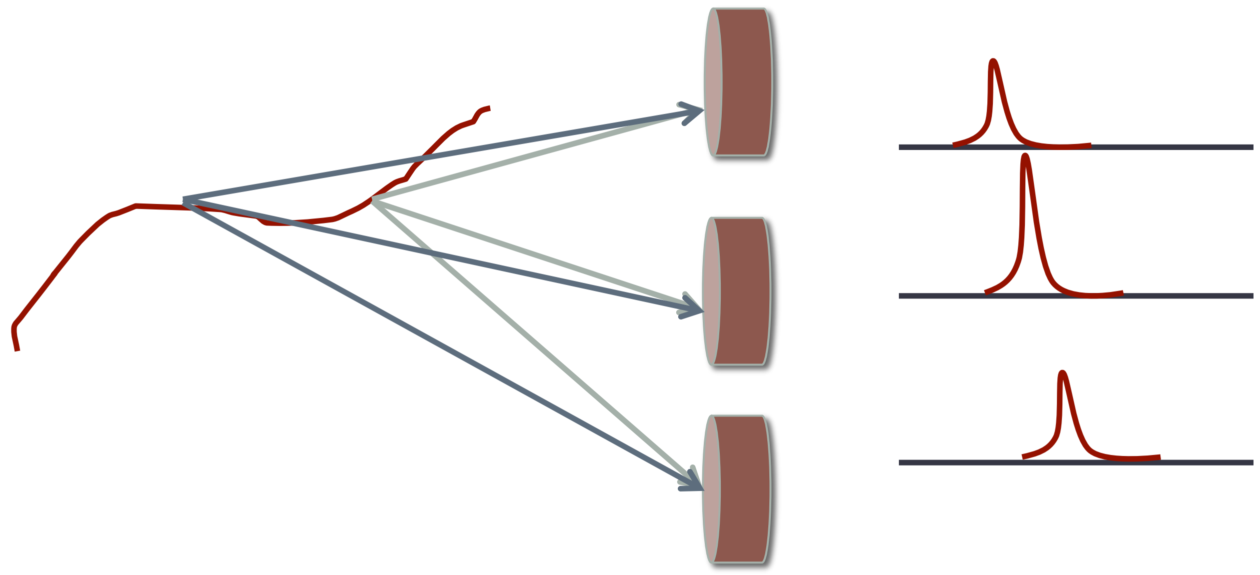
Geometrical T0 Finding

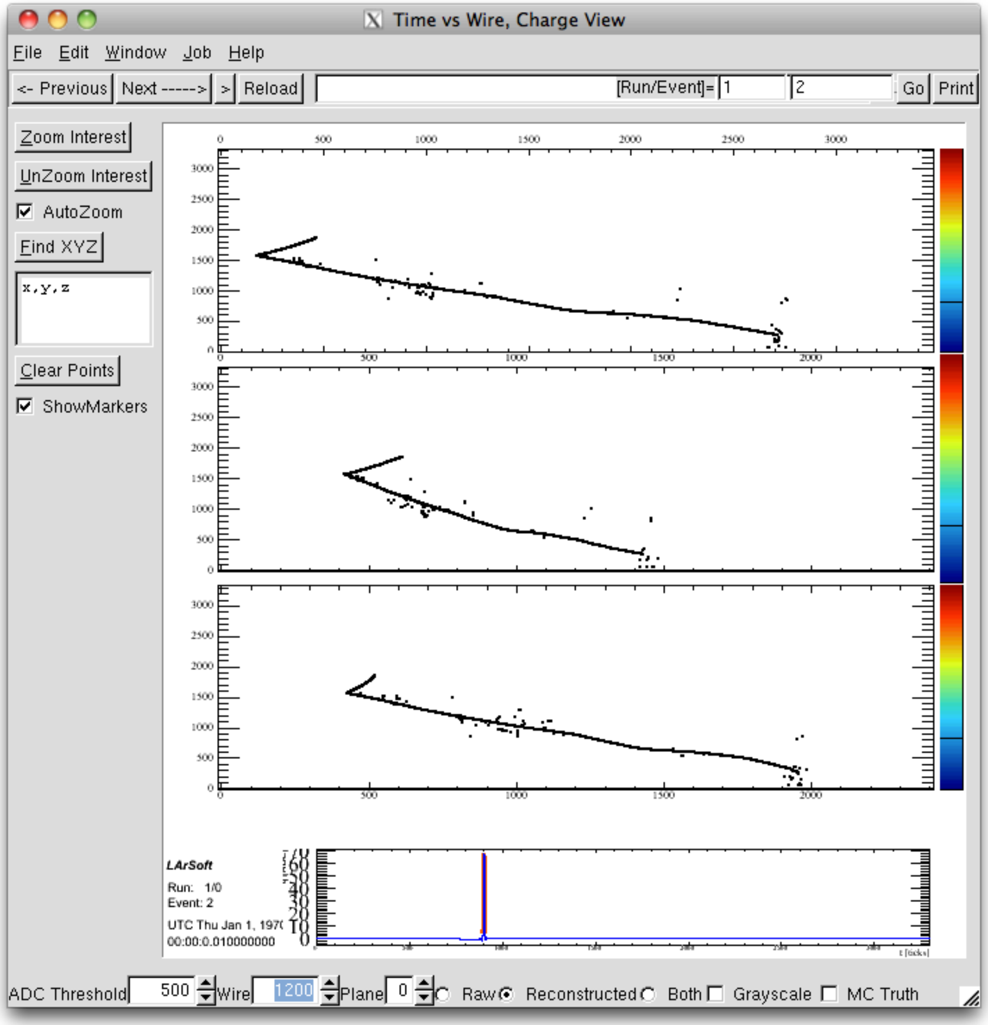
- **1: Find subevents by matching large PMT signals in time**
- **2: Make hypotheses of relative amount of light per PMT for each track**
- 3: Likelihood fit to match track to light hypothesis, and find T0.



TrackTimeAssoc

- Analyzer in the OpticalDetector package
- Make a quick hypothesis for the light from each track in the event per PMT
- Step along a bezier track in uniform intervals, querying the visibility at each point and multiplying by local dQdx.
- Light production can be dropped by quenching function. Visibility and quenching are both controlled by the PhotonVisibilityService





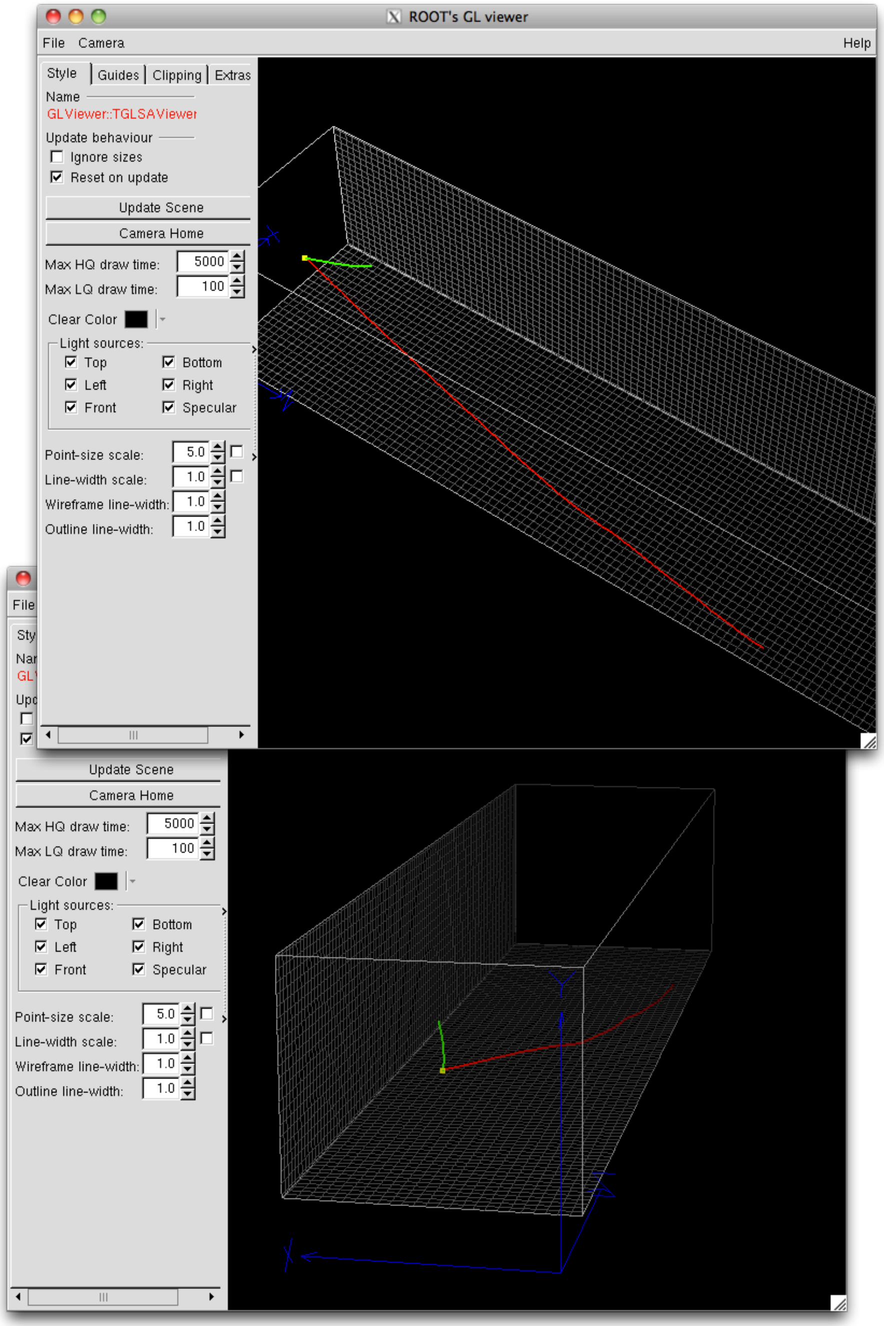
[Run/Event]= 1 2

[✓ AutoZoom]

[✓ ShowMarkers]

Run: 1/0
Event: 2
UTC Thu Jan 1, 1970
00:00:0.010000000

ADC Threshold 500 Wire 1200 Plane 0 Raw [✓ Reconstructed] Both Grayscale MC Truth



Name
GLViewer::TGLSAViewer

Update behaviour
Ignore sizes
[✓ Reset on update]

Max HQ draw time: 5000
Max LQ draw time: 100
Clear Color ■

Light sources:
[✓ Top] [✓ Bottom]
[✓ Left] [✓ Right]
[✓ Front] [✓ Specular]

Point-size scale: 5.0
Line-width scale: 1.0
Wireframe line-width: 1.0
Outline line-width: 1.0

- Arrival time prediction

Summary

- Review of developments since last time:
- Explicitly detector agnostic optical system simulation
- New geometry to allow optical detectors to be used consistently throughout LArSoft
- PhotonVisibilityService implemented to handle library jobs and interface with sim and reco optical methods
- New signal processing methods written for MicroBooNE style optical detectors
- Geometrical T0 reconstruction for multi track events well under way
- All LArSoft / MicroBooNE tools should be transferrable to LBNE with small to moderate effort.